AMENDMENTS TO THE CLAIMS

This listing of claims will replace all prior versions, and listings, of claims in the application:

1. (Currently Amended) A system comprising:

a Unix operating system;

a plurality of execution entities, the plurality of execution entities including a first execution entity;

an event control module adapted to create ~~an~~ event objects representing respective events each having a state, ~~one or more of the execution entities adapted to wait on the event~~ the first execution entity to wait on plural events;

a data structure associated with the first execution entity, the data structure containing information of the plural events that the first execution entity is waiting on, the data structure further containing an indicator settable to one of plural values to specify respective plural logical relationships between the plural events; and

a controller adapted ~~execution entities~~ to awaken the ~~one or more~~ first execution ~~entities~~ entity by signaling the first execution ~~entities if the~~ entity in response to one or more event state changes ~~to a predetermined state~~ of the states of the plural events according to the logical relationship specified by the indicator.

2. (Currently Amended) The system of claim 1, wherein the event control module is adapted to define ~~an event object representing the event, the event object associated with~~ a queue for a first one of the event objects, the queue having ~~one or more~~ plural entries corresponding to ~~one or more~~ plural execution entities waiting on the event represented by the first event object.

3. (Currently Amended) The system of claim 2, wherein the event control module is adapted to further create ~~one or more~~ second objects, wherein each entry of the queue comprises a link to a corresponding second object, each execution entity to sleep on an associated second object to wait on the event represented by the first event object.

4. (Original) The system of claim 3, wherein each second object is defined by a condition variable.

5. (Original) The system of claim 4, wherein the controller signals each thread by signaling the condition variable.

6. (Original) The system of claim 3, wherein each second object is defined by a condition variable and a mutex.

7. (Currently Amended) The system of claim ~~2~~ 1, wherein each event object contains an indication of the state of the event.

8. (Currently Amended) The system of claim 7, wherein the indication has a first state to indicate that the event has been signaled and a second state to indicate that the event has not been signaled~~, the predetermined state comprising the first state~~.

9. (Original) The system of claim 8, wherein each event object has a type indication to indicate whether the event object state indication is to be automatically reset to the second state from the first state once the event has been signaled or to be manually reset to the second state from the first state by an explicit action.

10. – 11. (Cancelled)

12. (Currently Amended) The system of claim ~~11~~ 1, further comprising queues associated with corresponding event objects representing events the first execution entity is waiting on, each queue containing an entry corresponding to the ~~one~~ first execution entity.

13. (Currently Amended) The system of claim 12, wherein the event control module is adapted to define a barrier object, the ~~one~~ first execution entity to sleep on the barrier object to wait on the plural events, the ~~entries of the queues each~~ queue of each event object containing a link to the barrier object.

14. (Original) The system of claim 13, wherein the barrier object is defined at least by a condition variable.

15. (Original) The system of claim 13, wherein the barrier object is defined at least by a condition variable and a mutex.

16. (Original) The system of claim 1, wherein the event control module comprises a library.

17. (Original) The system of claim 1, wherein the execution entities comprise threads.

18. (Original) The system of claim 17, further comprising plural processes, each process associated with one or more threads,

the event control module to create a local event to synchronize threads within a process and to create a global event to synchronize threads of different processes.

19. (Original) The system of claim 18, wherein the global event comprises a named event.

20. (Original) The system of claim 1, further comprising a plurality of nodes, each node comprising one or more of the plurality of execution entities.

21. (Currently Amended) An article comprising at least one storage medium containing instructions for providing event-based synchronization in a system in which execution entities are running, the instructions when executed causing the system to:

generate event objects in a Unix operating system environment representing events used for synchronizing execution entities in the system, each event object having a state to indicate if the corresponding event has been signaled;

provide ~~one or more~~ a queue containing entries associated with ~~the~~ a first event object, each entry associated with a corresponding execution entity, the plural entries of the queue enabling plural execution entities to wait on the first event object; and

selectively set a type variable to one of a first value and a second value, the first value indicating that the first event object is of an auto-reset type, and the second value indicating that the first event object is of a manual reset type;

in response to the state ~~of one~~ of the first event ~~objects~~ object indicating the corresponding event has been signaled, ~~use the one or more entries to signal one or more corresponding execution entities~~

automatically clear the state of the first event object to an un-signaled state and awaken only one of the plural execution entities waiting on the first execution object in response to the type variable being set to the first value, and

not clear the state of the first event object until manually cleared and awaken all threads waiting on the first event object in response to the type variable being set to the second value.

22. (Original) The article of claim 21, wherein the instructions when executed cause the system to further create barrier objects, each execution entity waiting on a corresponding barrier object to wait on an event.

23. (Currently Amended) The article of claim 22, wherein the instructions when executed cause the system to create barrier objects by defining each barrier object based on a condition variable according to [[a]] the Unix operating system.

24. (Currently Amended) The article of claim 22, wherein the instructions when executed cause the system to create barrier objects by defining each barrier object based on a condition variable and mutex according to [[a]] the Unix operating system.

25. (Currently Amended) The article of claim [[21]] 22, wherein ~~the instructions when executed cause the system to define a queue associated with each event object,~~ the queue ~~containing the one or more entries, the one or more~~ of the first event object contains entries pointing to the ~~one or more~~ barrier objects of the plural execution entities waiting on the first event object.

26. (Currently Amended) The article of claim 25, wherein the instructions when executed cause the system to provide a routine associated with each event object, the routine of the first event object to traverse ~~each~~ the queue of the first event object and to signal ~~one or more~~ the barrier objects pointed to by ~~one or more~~ the entries in the queue of the first event object.

27. (Cancelled)

28. (Currently Amended) A method of providing event-based synchronization in a system having plural execution entities, comprising:

providing one or more synchronization primitives;

defining a first object based on the one or more synchronization primitives;

defining [[an]] event ~~object~~ objects representing ~~an event~~ corresponding events, ~~the~~ each event object having a state to indicate whether the corresponding event ~~being~~ is signaled; ~~and~~

one of the execution entities sleeping on the first object to wait on the ~~event~~ events, the first object associated with a data structure containing information of the event objects, the data structure further containing an indicator settable to one of plural values to specify one of plural logical relationships between the events; and

awakening the one execution entity based on states of the event objects according to a logical relationship specified by the indicator.

29. (Currently Amended) The method of claim 28, further comprising signaling the first object in response to the states of event ~~object state indicating the event being signaled~~ objects according to the specified logical relationship to awaken the one execution entity.

30. (Cancelled)

31. (Currently Amended) The method of claim ~~30~~ 29, wherein providing the one or more synchronization primitives comprises providing one or more synchronization primitives defined in a Unix operating system.

32. (Original) The method of claim 31, wherein the one or more synchronization primitives comprises a condition variable, wherein signaling the first object comprises signaling the condition variable.

33. (Original) The method of claim 28, wherein providing the one or more synchronization primitives comprises providing one or more synchronization primitives defined in a Unix operating system.

34. (Currently Amended) The method of claim 28, further comprising ~~defining at least another event object representing another event,~~ the one execution entity ~~to add~~ adding entries to the event objects to enable the one execution entity to wait on plural events.

35. (Original) The method of claim 34, wherein the one execution entity adding the entries to the event objects comprises adding entries to queues associated with the event objects.

36. (Original) The method of claim 35, wherein adding the entries to the queues comprises adding a pointer to the first object.

37. (Original) A system comprising:

a Unix operating system;

a plurality of execution entities;

a storage module containing an event library; and

a processor adapted to execute the event library to provide an event-based synchronization mechanism comprising one or more events on which the plural execution entities are able to sleep.

38. (Original) The system of claim 37, further comprising plural processes, each process associated with one or more of the execution entities, wherein the synchronization mechanism comprises a local event synchronization mechanism to synchronize execution entities associated with one process, and a global event synchronization mechanism to synchronize execution entities associated with plural processes.

39. (New) The system of claim 1, wherein the indicator is settable to a first value to specify a logical AND relationship between the plural events, and in response to the first value of the indicator, the controller to awaken the first execution entity in response to all of the plural events waited on by the first execution entity being signaled.

40. (New) The system of claim 39, wherein the indicator is settable to a first value to specify a logical OR relationship between the plural events, and in response to the first value of the indicator, the controller to awaken the first execution entity in response to any of the plural events waited on by the first execution entity being signaled.

41. (New) The article of claim 21, wherein a first one of the execution entities waits on plural events represented by respective event objects, the instructions when executed causing the system to:

provide a data structure containing information of the plural events waited upon by the first execution entity, the data structure further containing an indicator settable to one of plural values to specify respective plural logical relationships between the plural events waited on by the first execution entity; and

awaken the first execution entity in response to states of the plural events waited upon by the first execution entity according to the logical relationship specified by the indicator.

42. (New) The article of claim 41, wherein the instructions when executed cause the system to set the indicator to a value to indicate a logical AND relationship, wherein awakening the first execution entity is in response to all of the plural events waited upon by the first execution entity being signaled.

43. (New) The method of claim 28, further comprising:

setting the indicator to a first value to specify a logical AND relationship between the events,

wherein awakening the one execution entity occurs in response to states of all the event objects being signaled, in response to the indicator being set to the first value.

44. (New) The article of claim 43, further comprising:

setting the indicator to a second value to specify a logical OR relationship between the events,

wherein awakening the one execution entity occurs in response to a state of any of the event objects being signaled, in response to the indicator being set to the second value.